2) Algoritmo da página 30

```
Stmt Level
  1    0   program pas002;
  1    0   (
               Dados 3 valores a,b,c verificar se eles podem
               ser valores dos lados de um triangulo e, se for,
               se e equilatero,isoceles ou qualquer.
                                                            }
  1    0    var a,b,c: integer; { valores a serem verificados }
  1    1    begin
  1    1       readln(a,b,c);
  2    1       if a< b+c and b < a+c and c < a+b
  3    1          then
  4    1             if a=b and b=c
  5    1                then writeln('triangulo equilatero')
  6    1                else if a=b or a=c or c=b
  7    1                        then writeln('triangulo isosceles')
  8    1                        else writeln('triangulo qualquer')
  9    1          else writeln('nao e triangulo');
 10    1   end.
```

```
 (execucao:)

A)pas002

34
56
23
triângulo qualquer
```


3) Algoritmo da página 76

```
Stmt Level
  1    0   program pas003;
  1    0   (
                Classificar um vetor numerico VET de 20 elementos em
                ordem crescente (metodo da bolha)
  1    0                                                    }
  1    0    type v = array[1..20] of integer;
  1    0    var vet : v;   { conjunto a ser classificado }
  1    1  *     aux,       { auxiliar para troca de elementos }
  1    1        bolha,     { mais alto elemento fora de ordem }
  1    1        lsup,      { tamanho do vetor a ser pesquisado }
  1    1        j : integer;          { aponta para o vetor }
  1    1    begin
  1    1        for j := 1 to 20 do
  2    2                begin
  3    3                    write('vet[',j:2,']' = ?');
  4    3                    readln(vet[j]);
  5    3        end;
  6    1        lsup := 20;
  7    1        while lsup > 1 do
  8    2           begin
  9    3              bolha := 0;
 10    3              for j := 1 to (lsup - 1) do
 11    4                  if vet[j] > vet[j+1]
 12    4                        then  { troca elem. j com elem. j+1 }
 13    4                            begin
 14    5                              aux := vet[j];
 15    5                              vet[j] := vet[j+1];
 16    5                              vet[j+1] := vet[j];
 17    5                              bolha := j;
 18    5                            end;
 19    3               lsup := bolha;  { ultima posicao trocada }
 20    3           end;
 21    1       for j := 1 to 20 do
 22    2           begin
 23    3               write('vet[');
 24    3               write(j:2);
 25    3               write('] = ');
 26    3               writeln(vet[j] : 5);
 27    3           end;
 28    1   end.
```